REMARKS

Reconsideration of the above-identified application, in view of the amendments and following remarks, is respectfully requested.

In the Official Action dated June 16, 2003, the Examiner first objected to the drawings under 37 C.F.R. §1.83(a) as allegedly not showing every feature of the invention specified in the claims. The Examiner is specifically referring to the limitation added in applicants' prior response (submitted April 23, 2003) directed to processing a bytecode sequence "..without modifying program bytecodes..". As will be explained in greater detail herein, in response, applicants have removed the offending language in each of independent Claims 1, 11 and 23 thus, obviating the rejection. The Examiner is respectfully requested to remove the objection to the drawings.

Further in the Office Action, the Examiner rejected independent Claims 1, 11 and 23 under 35 U.S.C. §112, first paragraph, as allegedly comprising subject matter not described in the specification in such a way as to reasonably convey to one skilled in the art that the inventor's had possession of the claimed invention at the time the application was filed. In light of the amendments to Claims 1, 11 and 23 removing language directed to processing of the bytecode sequence without modifying program bytecodes, Applicants respectfully submit that the rejection of Claims 1, 11 and 23 under 35 U.S.C. §112, first paragraph, is now obviated. Consequently, the Examiner is respectfully requested to withdraw the rejection of independent Claims 1, 11 and 23 under 35 U.S.C. §112, first paragraph.

Further, Applicants take this opportunity to amend the Specification on pages 6-7 and page 18 to correct a minor informalities in the text. Respectfully, no new matter is being submitted by this correction to the Specification.

Next, in the Official Action, the rejected Claims 1-3, 6-7, 9-13, 16-17 and 19-23 under 35 U.S.C. §103(a) as allegedly being unpatentable over Agesen (U.S. Patent No. 6,047,125) in view of Gosling (U.S. Patent No. 5,668,999) ("Gosling"). The Examiner further rejected Claims 4-5 and 14-15 under 35 U.S.C. §103(a) as allegedly being unpatentable over Agesen (U.S. Patent No. 6,047,125) ("Agesen I") in view of Agesen (U.S. Patent No. 5,909,579) ("Agesen II")). The Examiner further rejected Claims 8 and 18 under 35 U.S.C. §103(a) as allegedly being unpatentable over Agesen I in view of Gosling and further in view of O'Connor (U.S. Patent No. 6,098,089) ("O'Connor").

With respect to the Examiner's rejection of Claims 1-3, 6-7, 9-13, 16-17 and 19-23 under 35 U.S.C. §103(a) as allegedly being unpatentable over Agesen I in view of Gosling, Applicants respectfully disagree.

Particularly, in distinction to Agesen I whether taken alone or in combination with Gosling, the present invention is directed to a method for mapping a stack in a stack machine environment to help determine the shape of the stack at a given program counter. As now set forth in amended Claims 1, 11 and 23, this is accomplished by locating all start points possible for a given method, that is, at all of the entry points for the method and all of the exception entry points, and trying to find a path from the beginning of the method to the program counter in question, and then iteratively processing the sequence of bytes at each branch until the destination program counter is reached. Once the path is found, a simulation is run of the stack through that path, which is used as the virtual stack for the purposes of the garbage collector.

Unique to the invention is the notion that the path mapping stops, i.e. the path is complete, when the destination PC is reached for the first time. That is, there may be more than one path to get from the beginning of the method to the destination PC, but according to the invention, the mapping may stop as soon as the first path is established. Accordingly, each of independent Claims 1, 11 and 23 are being amended to set forth a method step for mapping a valid stack up to a destination program counter including mapping a path of control flow on the stack from any start point in a selected method to the destination program counter by locating a linear path from the beginning of the method to the destination program counter and iteratively processing <u>an existing</u> bytecode sequence at each branch<u>, and identifying the path as complete when</u> the destination counter is reached. That is, the purpose of the present invention, unlike Agesen I, is not to modify bytecodes but to simplify and make more economical the preparation for garbage collection which is beneficial and neither addressed nor suggested in the cited prior art. Thus it happens that virtually the only reason to perform this with current and probably future Java machines is to facilitate garbage collections (See discussion in the first full paragraph of the prior art section of Page 6 of the present application). If there were other reasons to produce a stack map, however, the invention would be applicable.

Applicants' note a further amendment to the Claims 1, 11 and 23 that set forth that <u>an existing</u> bytecode sequence for each branch is iteratively processed until the destination program counter is reached. Respectfully this is intended to imply that the <u>bytecode is not modified by the invention</u> as the specification does not anywhere describe that the bytecodes in the target program are ever modified. While bytecodes may have been modified in the past, it is not because of the invention (i.e., it could have been modified or not (depending on what the baseline was)).

Respectfully, no new matter is being entered by the amendment to the Claims 1, 11 and 23 setting forth the step of identifying the path as complete when the destination counter is reached as page 7, amended lines 13-16 set forth "The processing is repeated iteratively, starting from the beginning of the method and then from each branch target until the destination program counter has been processed." Further, in Figure 1A, decision block 108 determines whether the destination PC has been seen, and if so, the path has been completed and the system creates at block 110 the reverse map of the path. This means that when even one path to the destination PC has been walked, the method of the invention will stop the path-mapping phase of the process.

Respectfully, none of the cited prior art documents discloses analyzing only enough bytecodes to establish a path from the beginning of a program or method, as clearly provided in the subject amended claims. The shortened process is in contrast to Agesen I, which describes walking through the entire sequence of instructions, not stopping until every instruction has been analyzed for conflicts (col. 5, lines 30-37, col. 5, lines 40-43, col. 9, lines 54-57, col. 9, line 66 to col. 10, line 2 and Figure 5, blocks 510, 530). Similarly, Gosling discloses processing all the instructions in a program (col. 7, lines 33-35 and Figure 4A, block 410). As to Agesen II, it discloses analyzing each bytecode (col. 7, lines 47-52) and storing the live pointer information for each bytecode (col. 7, lines 35-37, col. 9, lines 24-28 and Figure 4, block 408). Thus, none of the cited references whether taken alone or in combination, teaches or suggests the ability for terminating the path-mapping phase of the process upon reaching the destination counter.

Thus, advantageously, by the system and method of the present invention, efficiencies are created in terms of processing time and memory allocations by providing the ability to terminate the path-mapping phase of the process upon reaching a specified program counter (PC) during program execution.

As stated in applicants' prior response of April 23, 2003 (entered upon the filing of a RCE on May 21, 2003), it is respectfully submitted that the Agesen I does not anticipate the invention as set forth in amended Claims 1 and 11. Agesen I essentially is directed to a method for modifying a sequence of instructions to improve memory management within a storage device during execution of the instructions, which comprises steps, performed by a processor, of (a) analyzing the sequence of instructions for a conflict indicating an undeterminable variable type, (b) determining the type of conflict, and (c) modifying the sequence of instructions to eliminate the conflict based on the determination. Further, as stated in the Summary (Col. 5, lines 16 et seq. of Agesen I patent) the subject matter of Agesen is to improve garbage collection by modifying code that introduces a conflict in the assignment of variables for identifying references. The conflict exists, for example, when a method includes code defining at least two control paths leading to a common subroutine and the garbage collector initiated during execution of the subroutine cannot determine whether a variable represents a reference. By rewriting the code in Agesen I the conflicts are eliminated.

With respect to the Examiner's rejection of Claims 4-5 and 14-15 under 35 U.S.C. §103(a) as allegedly being unpatentable over Agesen I in view of Agesen II, Agesen II describes, as stated in the summary of the invention at col. 3, line 63 to col.4, line 17, a method in which:

> "...live pointer information for a stream of bytecodes is precomputed for each bytecode. The precomputed full live pointer information is stored only for bytecodes at predetermined intervals in the stream. Between the bytecodes for which full live pointer information is stored, changes in the live pointer information produced by each bytecode are encoded using a suitable compressive coding and stored. Later, when garbage collection is initiated, the full live pointer information for the nearest bytecode preceding the desired bytecode boundary is retrieved along with the intervening coded changes. The changes are decoded and applied to the retrieved live pointer information to generate the live pointer information at the desired bytecode boundary. In accordance with one embodiment of the invention, the live pointer changes are delta encoded so that each code contains information relating to the live pointer changes produced by a bytecode from the live pointer information as modified by the previous delta code. In accordance with another embodiment of the invention, the delta coded changes are encoded with a Huffman encoding scheme.."

The basic invention in Agesen II is choosing to store full information for only every *n*th bytecode in the stream, and recreating the deltas in between them. The Examiner states that Agesen II "teaches a 'bytecode analyzer mechanism' which determines the changes which the bytecode makes to the live pointer locations (col.8, lines 20-24)." He further states that "the system has breakpoints or computation stops at bytecode boundaries for determining live pointer information (col.3, lines 10-13)."

The applicants respectfully submit that Agesen I and Agesen II combined do not disclose either mapping a path of control flow or simulating stack actions including iteratively processing <u>an existing</u> bytecode sequence at each branch, <u>and identifying the path as complete when</u> the destination counter is reached, as discussed above with respect to amended Claims 1 and 11, from which claims 4, 5, 14 and 15 depend. Neither do they disclose any steps that are similar to, or that suggest the use of, those two elements. Thus since the two references do not contain or suggest all the elements of the claims, the applicants submit respectfully that the invention of Claims 4, 5, 14 and 15 is not obvious by the combined references and the Examiner is respectfully requested to withdraw the rejections of Claims 4-5 and 14-15 under 35 U.S.C. §103(a).

With respect to the rejection of Claims 6-7, 9, 16-17 and 19 under 35 U.S.C. §103(a) as allegedly being unpatentable over Agesen I in view of Gosling, the Examiner states that Gosling supplies, col.5, lines 21-29, the step of generating a virtual stack that is omitted from Agesen I. However, Claims 1 and 11, from which Claims 6 and 16 respectively depend, are restricted to mapping a path of control flow only from an arbitrary start point up to a destination program counter, and <u>identifying the path as complete when</u> the destination counter is reached. Therefore the virtual stack thus created is only a stack for part of the program, unlike Gosling which creates a virtual stack for the entire program, i.e., "used in the same way as a regular stack…", (see col.5, lines 35-40 of Gosling). That is, the method of the present invention eliminates the time and cost of generating a complete virtual stack in the manner of Gosling. Further, as the limitations of amended Claims 1 and 11 have been discussed above, and their distinctions over the cited prior art have been pointed out, it is submitted that the combined references of Agesen1 and Gosling do not include any similar steps that would suggest all the steps of Claims 6 and 16, which depending from amended Claims 1 and 11, respectively. The applicant respectfully submits that Claims 6 and 16 have not been shown to be obvious over the cited art and respectfully requests that the rejection of Claims 6 and 16 under 35 U.S.C. §103(a) be withdrawn.

The same arguments may be applied to Claims 7 and 17, these each depend respectively from Claims 6 and 16, and include the limitations of the parent claims as discussed above.

It is respectfully submitted that Claims 7 and 17 are thus allowable also. The